Blockchain As A Service Application a l'IoT

J. BRIFFAUT – JEREMY.BRIFFAUT@INSA-CVL.FR

INSA-CVL

LIFO/SDS

Présentation du LIFO

5 équipes :

- CA : **Contraintes et Apprentissage**
- GAMoC : **GAMoC : Graphes, Algorithmes et Modèles de Calcul**
- LMV : **Langages, Modèles et Vérification**
- PaMDA : **ProgrAmmation, Modélisation et vérification D'Applications parallèles et distribuées**
- SDS : **Sécurité des Données et des Systèmes**
  - **Le contrôle d'accès et le contrôle d'usage**
  - **Les modèles d'attaque sur les données et les algorithmes**
  - **Les approches formelles de la sécurité et de la vie privée**
  - **Les applications pluridisciplinaires de la sécurité informatique**

Projets en cours

- FUI ATELYN

trAçabiliTE Longitudinale hYbride uNitaire

- IOT-CIA

IOT Confidentiality-integrity-and-availability

Plan

I – Openstack

II - Multichaind

Openstack

« OpenStack est un ensemble de logiciels open source permettant de déployer des infrastructures de cloud computing (infrastructure en tant que service). La technologie possède une architecture modulaire composée de plusieurs projets corrélés (Nova, Swift, Glance...) qui permettent de contrôler les différentes ressources des machines virtuelles telles que la puissance de calcul, le stockage ou encore le réseau inhérents au centre de données sollicité.Le projet est porté par la Fondation OpenStack, une organisation non-commerciale qui a pour but de promouvoir le projet OpenStack ainsi que de protéger et d'aider les développeurs et toute la communauté OpenStack1.De nombreuses entreprises ont rejoint la fondation OpenStack2,3. Parmi celles-ci on retrouve : Canonical, Red Hat, SUSE, eNovance, AT&T, Cisco, Dell, HP, IBM, Yahoo!, Oracle4, Orange, Cloudwatt, EMC, VMware, Intel, OVH, NetApp. »

https://fr.wikipedia.org/wiki/OpenStack

Images provenant de :

https://docs.openstack.org/liberty/networking-guide/

https://docs.openstack.org/admin-guide/common/

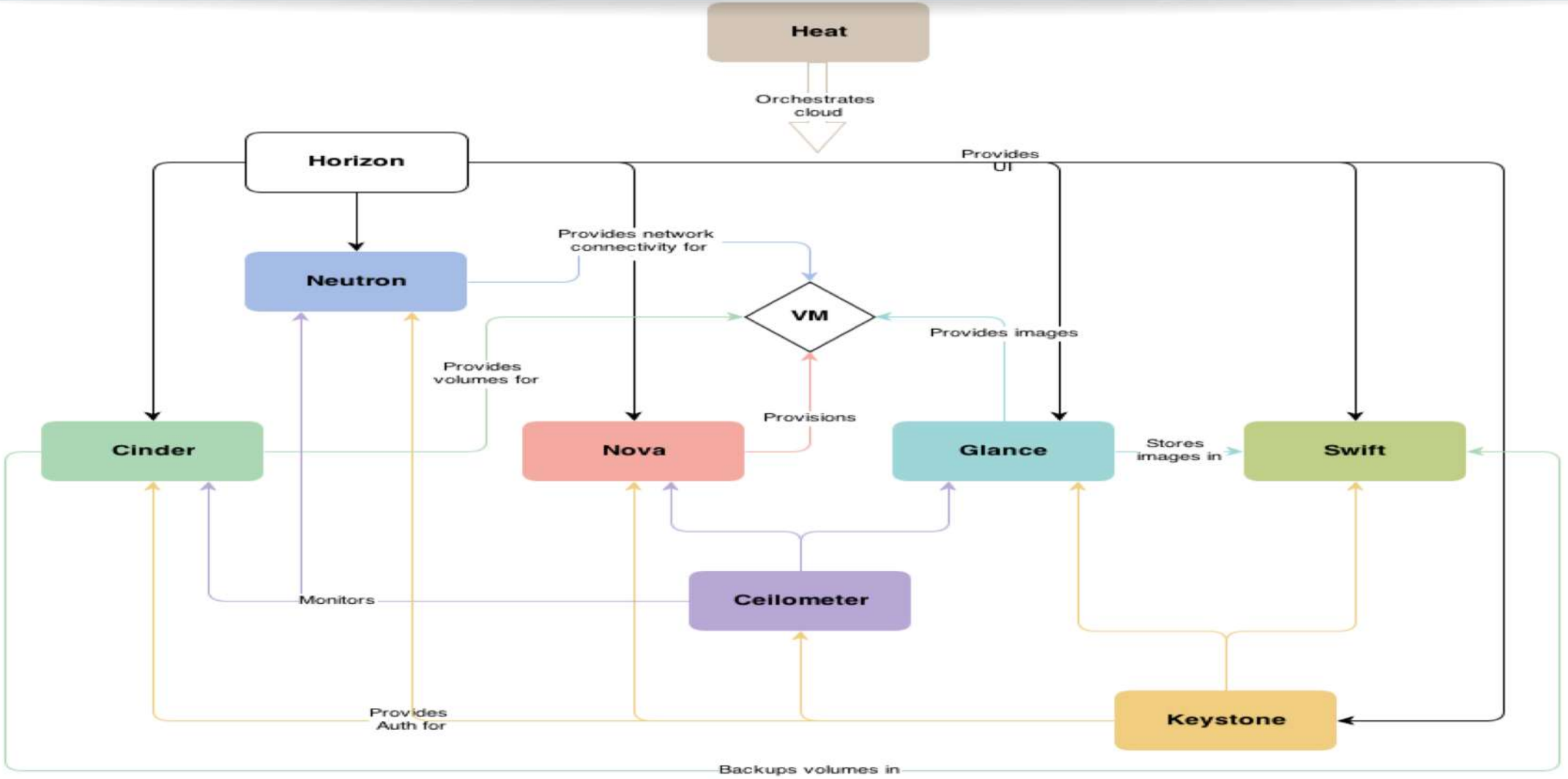
Openstack - en résumé

- Permet de mettre en place un Cloud
  - Open-source, basé sur Linux (je conseille CENTOS), gratuit
  - Connection possible avec VMWare ESX
- 3 types de nœud :
  - **Controller** : gère l'architecture (orchestrateur), héberge le frontend (WEB-ui + Command-line UI)
  - **Network** : fournit la fonctionnalité réseau (VM<->VM , VM<->EXT)
    - Basé sur openvswitch
  - **Compute** : héberge les Machines Virtuelles (instance ou VM), i.e un hyperviseur
    - Basé sur KVM/Libvirt
- Architecture minimale (classique)
  - 1 controller, 1 network, X compute
- Architecture modulaire
  - 1 composant = 1 service = 1 projet = 1 nom
  - Beaucoup de composants imbriqués => **complexité**

Composants/Projets Openstack

Service	Project name	Description
Dashboard	Horizon	Provides a web-based self-service portal to interact with underlying OpenStack services, such as launching an instance, assigning IP addresses and configuring access controls.
Compute	Nova	Manages the lifecycle of compute instances in an OpenStack environment. Responsibilities include spawning, scheduling and decomissioning of virtual machines on demand.
Networking	Neutron	Enables network connectivity as a service for other OpenStack services, such as OpenStack Compute. Provides an API for users to define networks and the attachments into them. Has a pluggable architecture that supports many popular networking vendors and technologies.
Storage		
Object Storage	Swift	Stores and retrieves arbitrary unstructured data objects via a RESTful, HTTP based API. It is highly fault tolerant with its data replication and scale out architecture. Its implementation is not like a file server with mountable directories.
Block Storage	Cinder	Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the creation and management of block storage devices.
Shared services		
Identity service	Keystone	Provides an authentication and authorization service for other OpenStack services. Provides a catalog of endpoints for all OpenStack services.
Image Service	Glance	Stores and retrieves virtual machine disk images. OpenStack Compute makes use of this during instance provisioning.
Telemetry	Ceilometer	Monitors and meters the OpenStack cloud for billing, benchmarking, scalability, and statistical purposes.
Higher-level services		
Orchestration	Heat	Orchestrates multiple composite cloud applications by using either the native HOT template format or the AWS CloudFormation template format, through both an OpenStack-native REST API and a CloudFormation-compatible Query API.
Database Service	Trove	Provides scalable and reliable Cloud Database-as-a-Service functionality for both relational and non-relational database engines.

Architecture

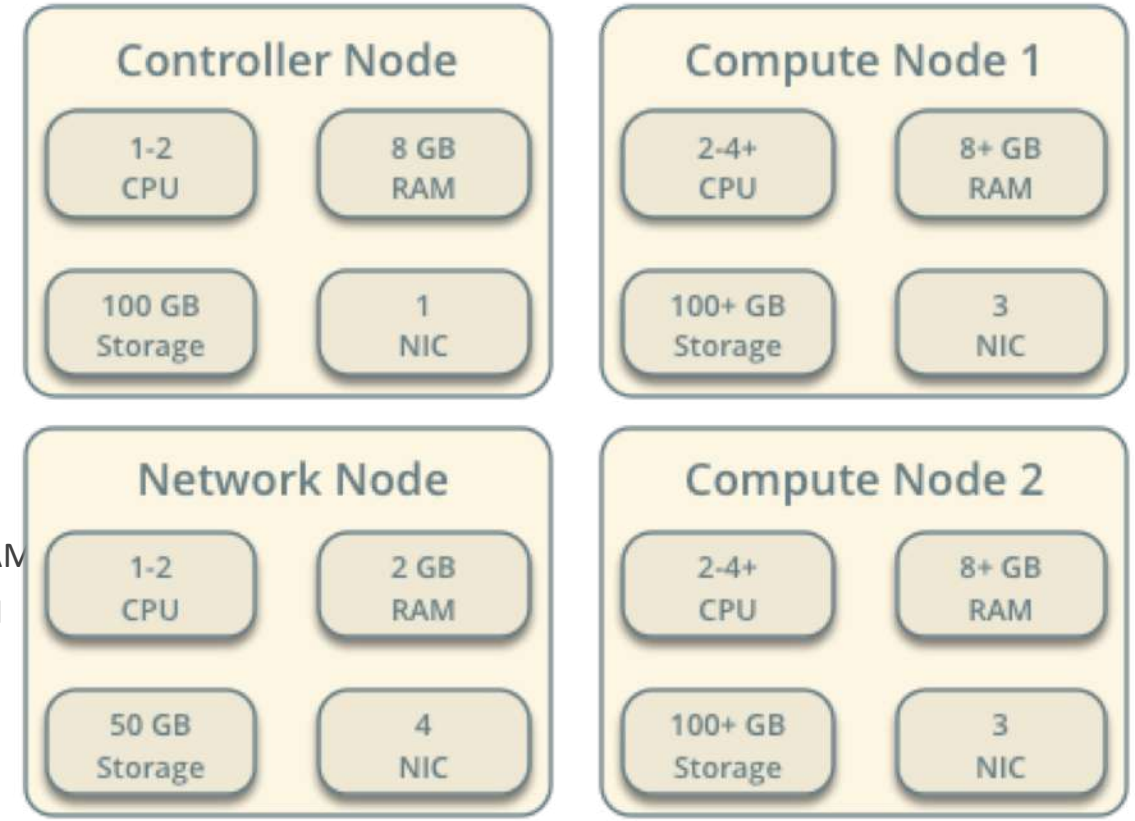


Openstack - Hardware

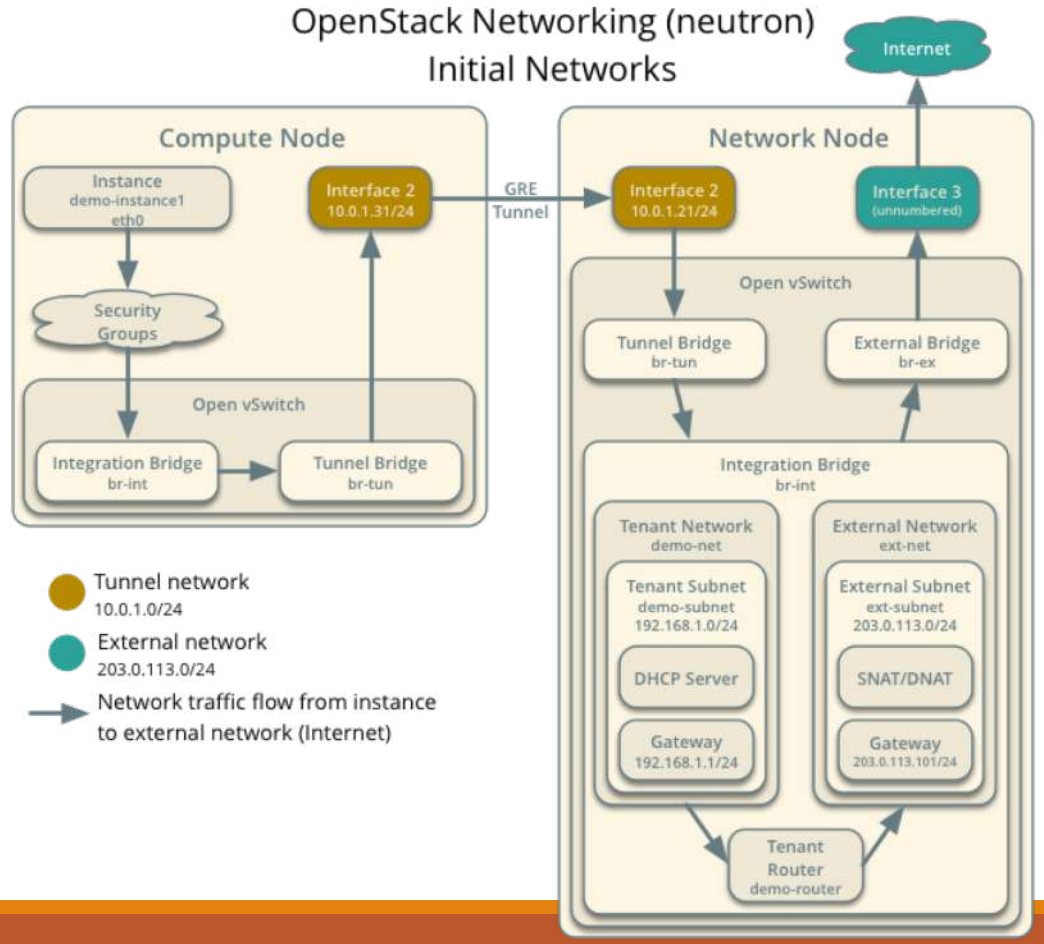
Minimal requis :

- 2 serveurs
  - 1 controller+network
    - 2 interfaces réseau (interne+externe)
  - 1 compute (X cœurs = X Vms)
    - 1 interface (interne)

- Mise en place a l'INSA-CVL
  - 6x DELL T110 – Xeon X3450 @ 2.67GHz - 8Go RAM
  - 6x DELL T110 II – Xeon E3-1240 V2 @ 3.40GHz - 16GO RAM
  - 4x DELL T130 – Xeon E3-1220 v5 @ 3.00GHz - 16GO RAM



Openstack - reseau



Openstack – Déploiement automatique

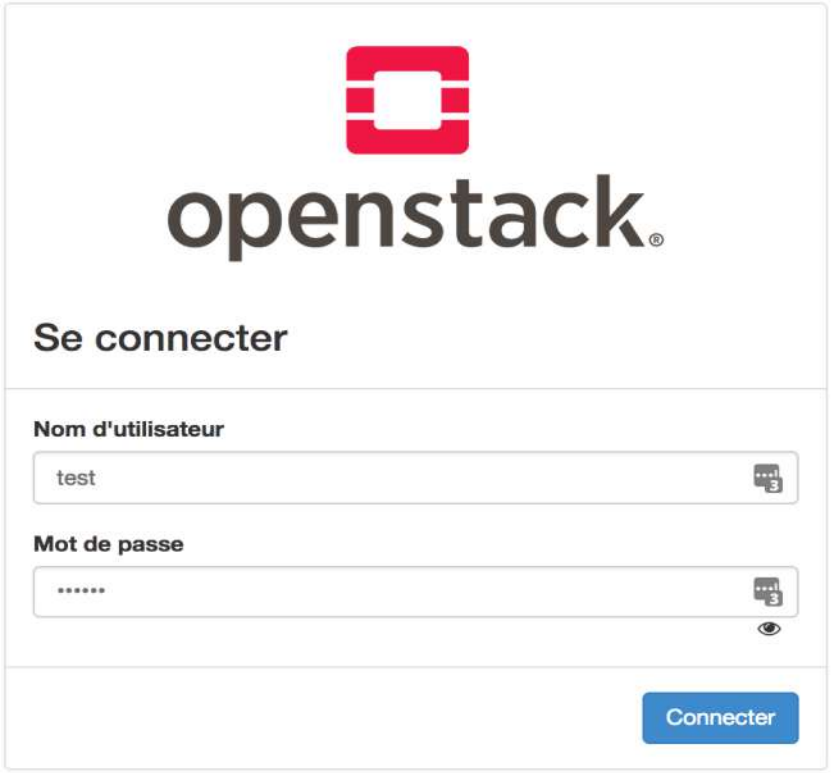
- Utilisation via PACKSTACK
  - Déploiement Openstack (sur CENTOS 7) automatique via PUPPETS
  - Le tout en 5 commandes (mais il faut attendre 20-30 minutes)

```
yum install -y https://www.rdoproject.org/repos/rdo-release.rpm
yum install -y centos-release-openstack-ocata
yum update -y
yum install -y openstack-packstack
packstack --answer-file=/root/answer.txt
```

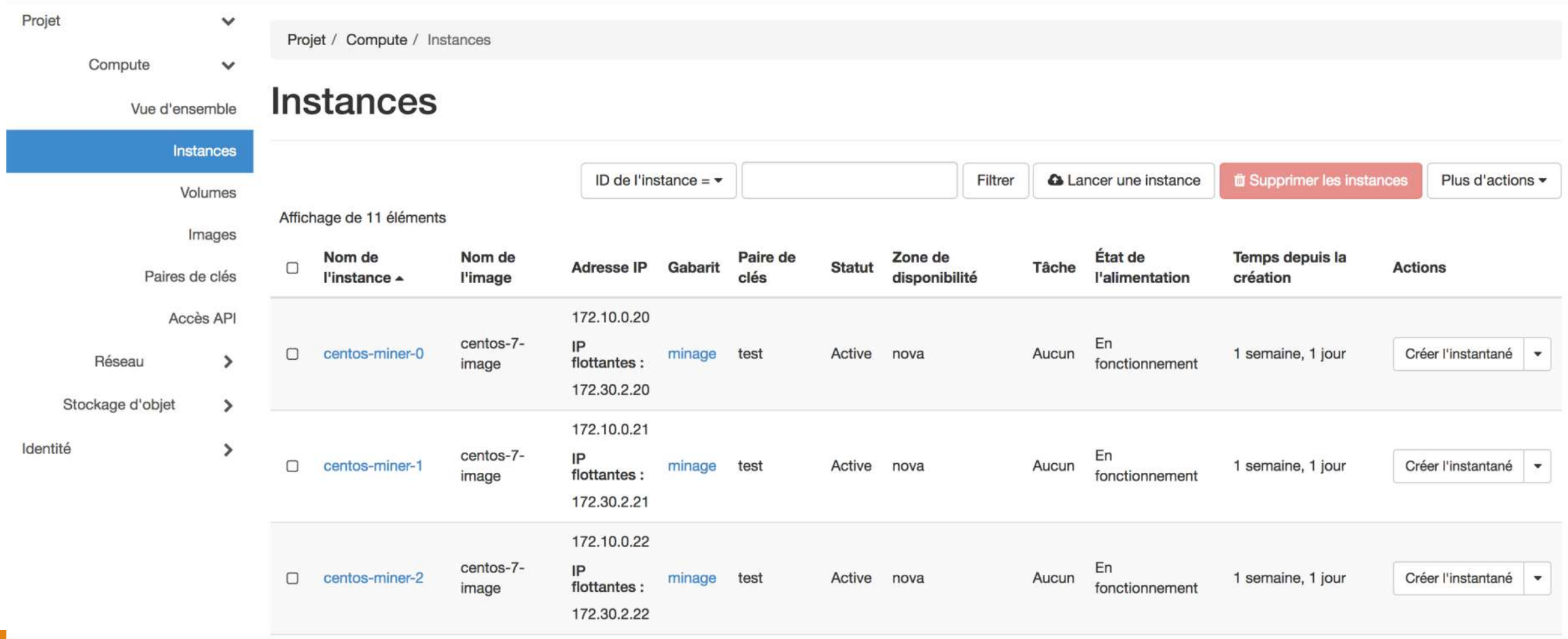
  - Le plus difficile étant de trouver les bons paramètres pour le fichier de configuration (answer.txt)
  - Il faut ensuite paramétrer Openstack
    - Utilisation des commandes clientes en mode texte, par exemple :

```
neutron net-create external_network --provider:network_type flat --provider:physical_network extnet  --router:external
neutron subnet-create --name public_subnet --enable_dhcp=False --allocation-pool=start=172.30.2.10,end=172.30.2.250 \
        --gateway=172.30.2.254 external_network 172.30.2.0/24
curl http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-x86_64-disk.img | glance \
    image-create --name='cirros image' --visibility=public --container-format=bare --disk-format=qcow2
```

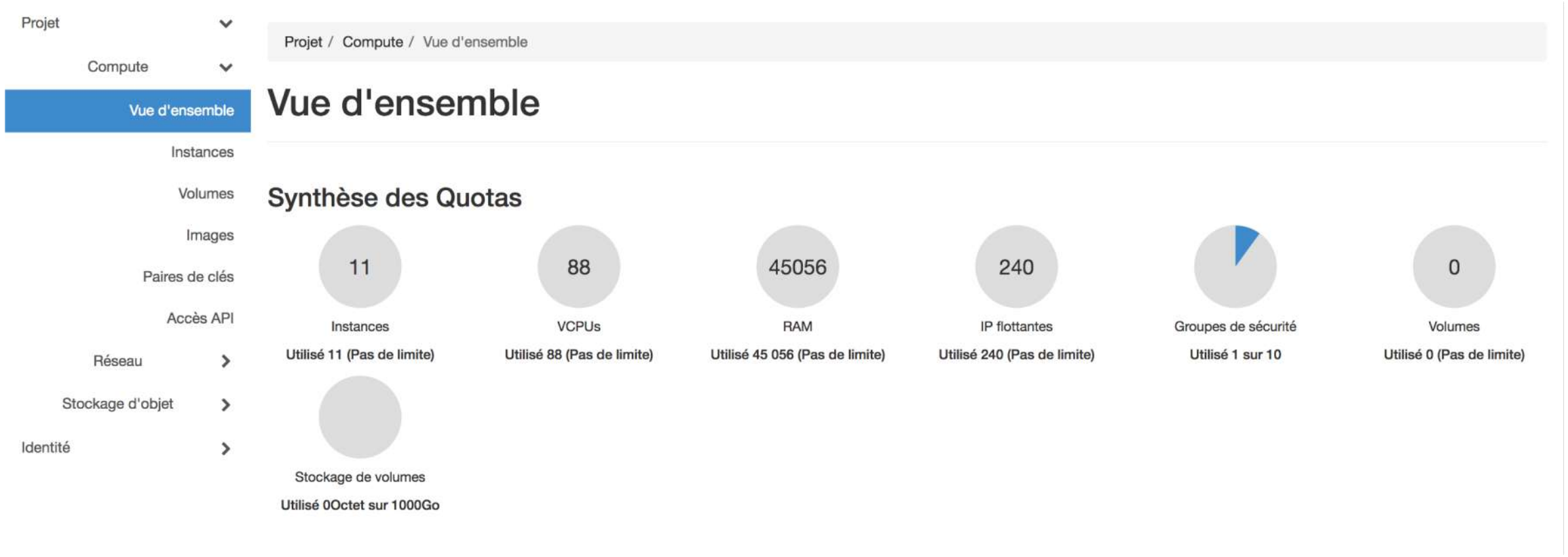
Openstack - dashboard



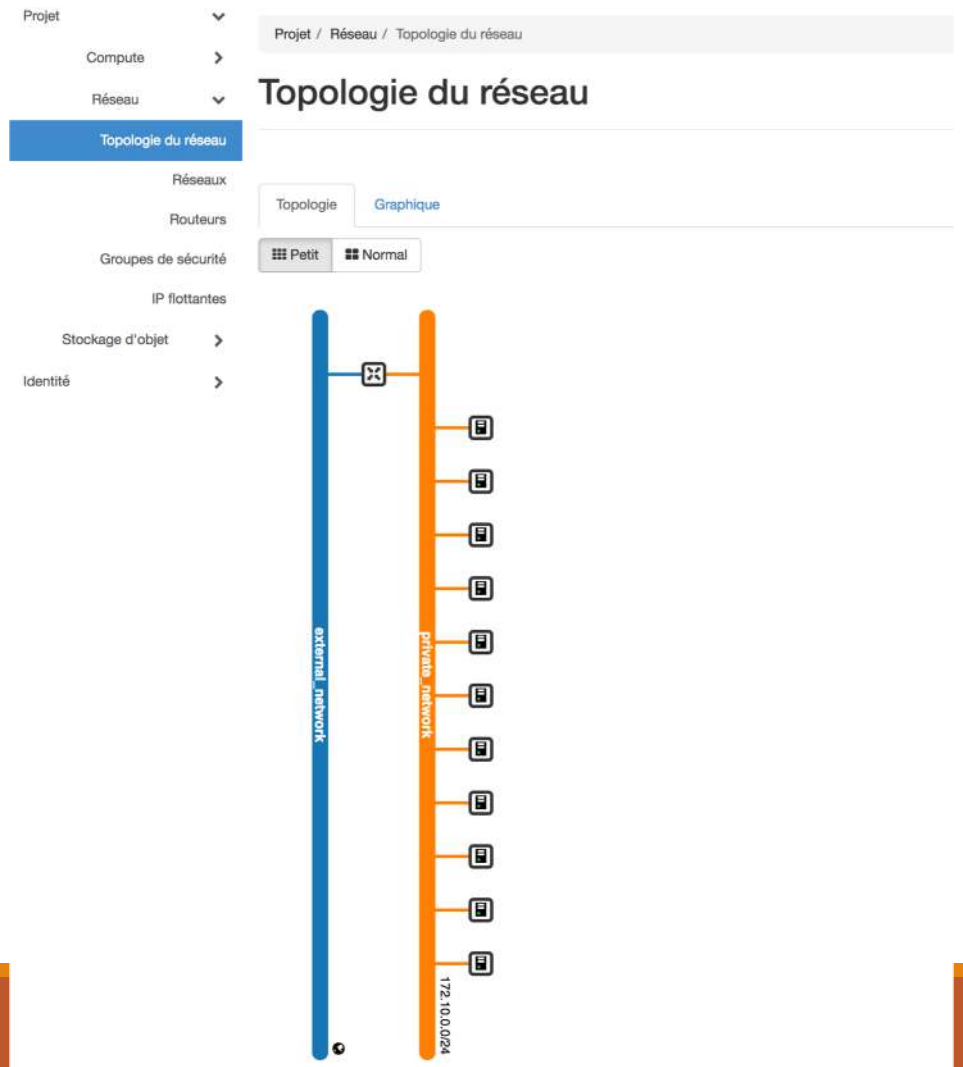
Openstack – dashboard – liste des VMs



Openstack – dashboard– vue d’ensemble



Openstack–dashboard–Topologie réseau



Plan

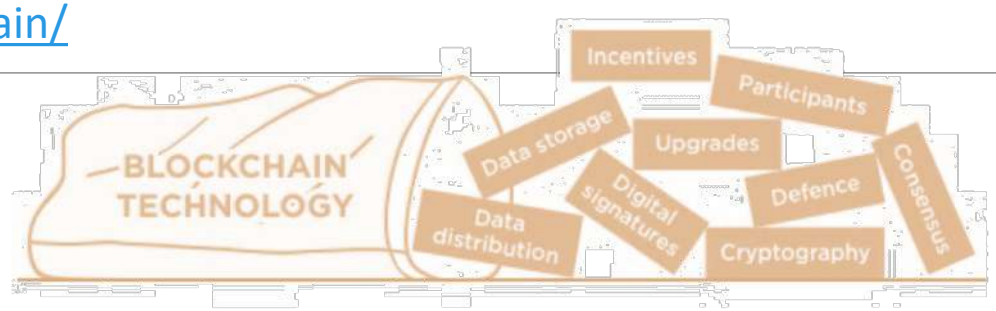
I – Openstack

II - Multichaind

Blockchain ?

http://blockchainmtl.com/quest-ce-que-la-blockchain/

Blockchain est un peu plus que :
•**Distributed** •**Secure** •**Logfile**



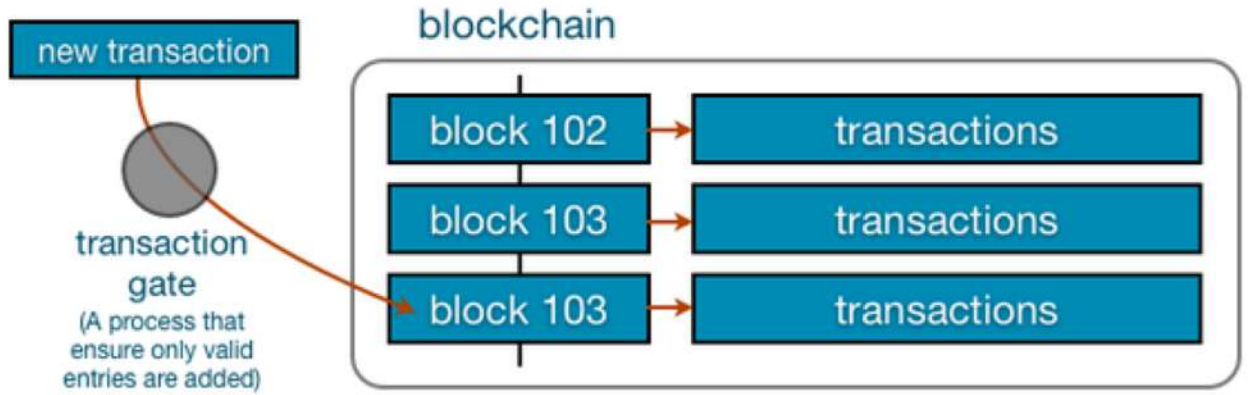
Registre décentralisé qui enregistre les transferts de valeur. Chaque transaction est cryptographiquement chainée à la précédente.

Latest Transactions		
971a19c7b41cfcf165cd9023f...	< 1 minute	0.15972025 BTC
753c058eff52c85e3b3b982a4...	< 1 minute	0.0143 BTC
9647d15115d5c1d7ccf8f3dc9...	< 1 minute	2.67254865 BTC
51ceed2953915fcc722c66496...	< 1 minute	48.83591155 BTC

Blockchain

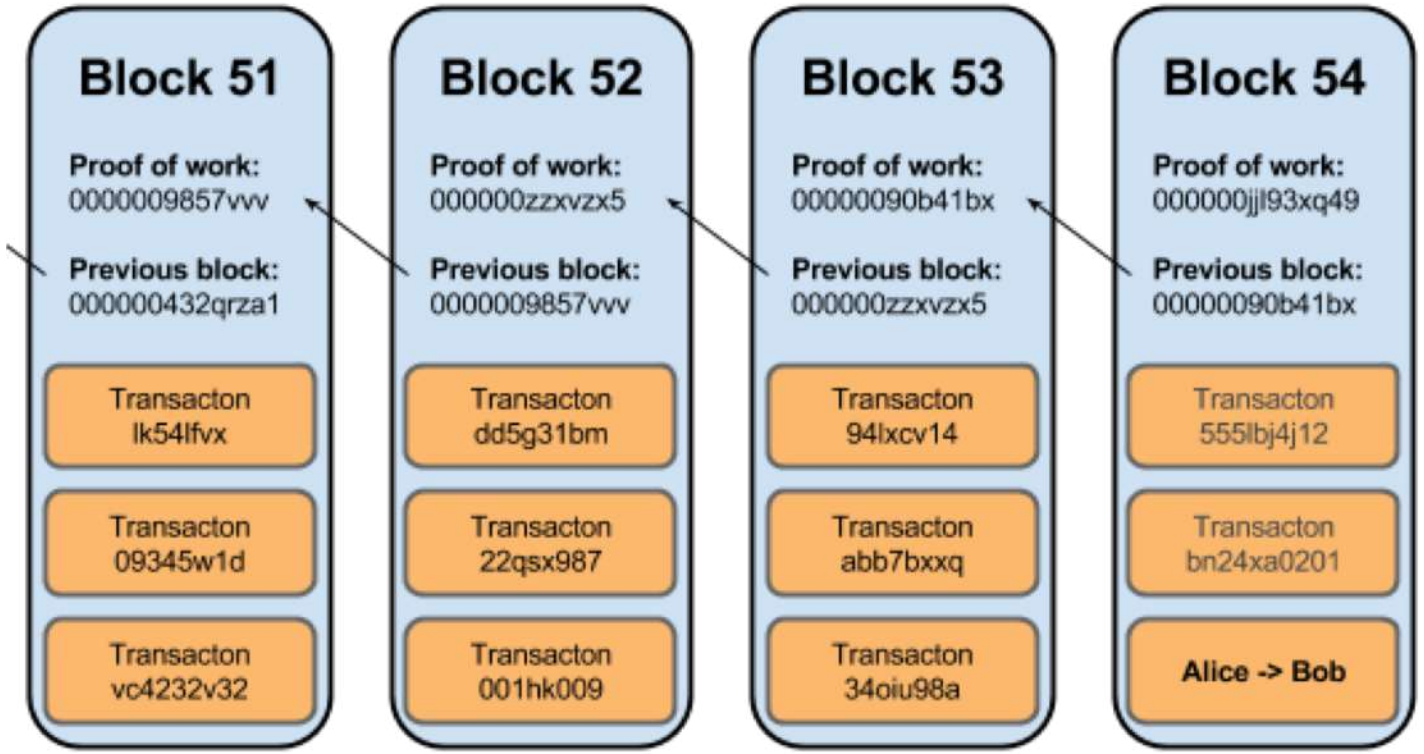
•Un protocole qui propose une gestion de “biens” numériques decentralisée, pseudo-anonyme, pair à pair

•Un registre publique de transactions liées via des blocs



Transactions aren’t recognized until they are added to the blockchain. Tampering is immediately evident, and the blockchain is safe as record because everyone has a copy. The source of discrepancies is also immediately obvious.

Chainage de block



How a ₿itcoin transaction works

Bob, an online merchant, decides to begin accepting bitcoins as payment. Alice, a buyer, has bitcoins and wants to purchase merchandise from Bob.

WALLETS AND ADDRESSES

Bob and Alice both have Bitcoin "wallets" on their computers.

Wallets are files that provide access to multiple Bitcoin addresses.

An address is a string of letters and numbers, such as 1HULMwZEPkjEPeCh43BeKJLTybLCWrfDpN.

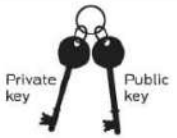
CREATING A NEW ADDRESS

Bob creates a new Bitcoin address for Alice to send her payment to.

Each address has its own balance of bitcoins.

SUBMITTING A PAYMENT

Alice tells her Bitcoin client that she'd like to transfer the purchase amount to Bob's address.



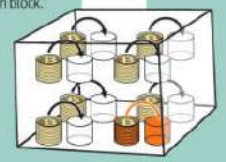
Public Key Cryptography 101
When Bob creates a new address, what he's really doing is generating a "cryptographic key pair," composed of a private key and a public key. If you sign a message with a private key (which only you know), it can be verified by using the matching public key (which is known to anyone). Bob's new Bitcoin address represents a unique public key, and the corresponding private key is stored in his wallet. The public key allows anyone to verify that a message signed with the private key is valid.

It's tempting to think of addresses as bank accounts, but they work a bit differently. Bitcoin users can create as many addresses as they wish and in fact are encouraged to create a new one for every new transaction to increase privacy. So long as no one knows which addresses are Alice's, her anonymity is protected.

VERIFYING THE TRANSACTION

Gary, Garth, and Glenn are Bitcoin miners.

Their computers bundle the transactions of the past 10 minutes into a new "transaction block."



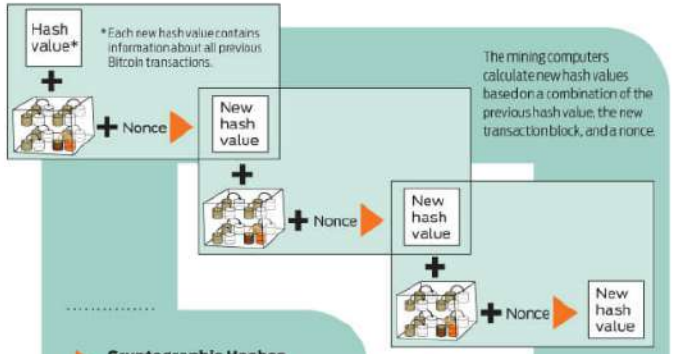
The miners' computers are set up to calculate cryptographic hash functions.

Private key

Public key

Alice's wallet holds the private key for each of her addresses. The Bitcoin client signs her transaction request with the private key of the address she's transferring bitcoins from.

Anyone on the network can now use the public key to verify that the transaction request is actually coming from the legitimate account owner.



The mining computers calculate new hash values based on a combination of the previous hash value, the new transaction block, and a nonce.

Cryptographic Hashes
Cryptographic hash functions transform a collection of data into an alphanumeric string with a fixed length, called a hash value. Even tiny changes in the original data drastically change the resulting hash value. And it's essentially impossible to predict which initial data set will create a specific hash value.

The root of all evil	6d0a1899 086a... (56 more characters)
The root of all evil	485c 6be4 6dde...
The root of all veil	b8db 7ee9 8392...

Nonces
To create different hash values from the same data, Bitcoin uses "nonces." A nonce is just a random number that's added to data prior to hashing. Changing the nonce results in a wildly different hash value.

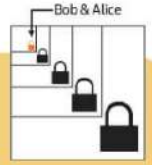
The root of all evil???	0000 0000 0000 ...

Creating hashes is computationally trivial, but the Bitcoin system requires that the new hash value have a particular form—specifically, it must start with a certain number of zeros.

The miners have no way to predict which nonce will produce a hash value with the required number of leading zeros. So they're forced to generate many hashes with different nonces until they happen upon one that works.

Each block includes a "coinbase" transaction that pays out 50 bitcoins to the winning miner—in this case, Gary. A new address is created in Gary's wallet with a balance of newly minted bitcoins.

TRANSACTION VERIFIED



As time goes on, Alice's transfer to Bob gets buried beneath other, more recent transactions. For anyone to modify the details, he would have to redo the work that Gary did—because any changes require a completely different winning nonce—and then redo the work of all the subsequent miners. Such a feat is nearly impossible.

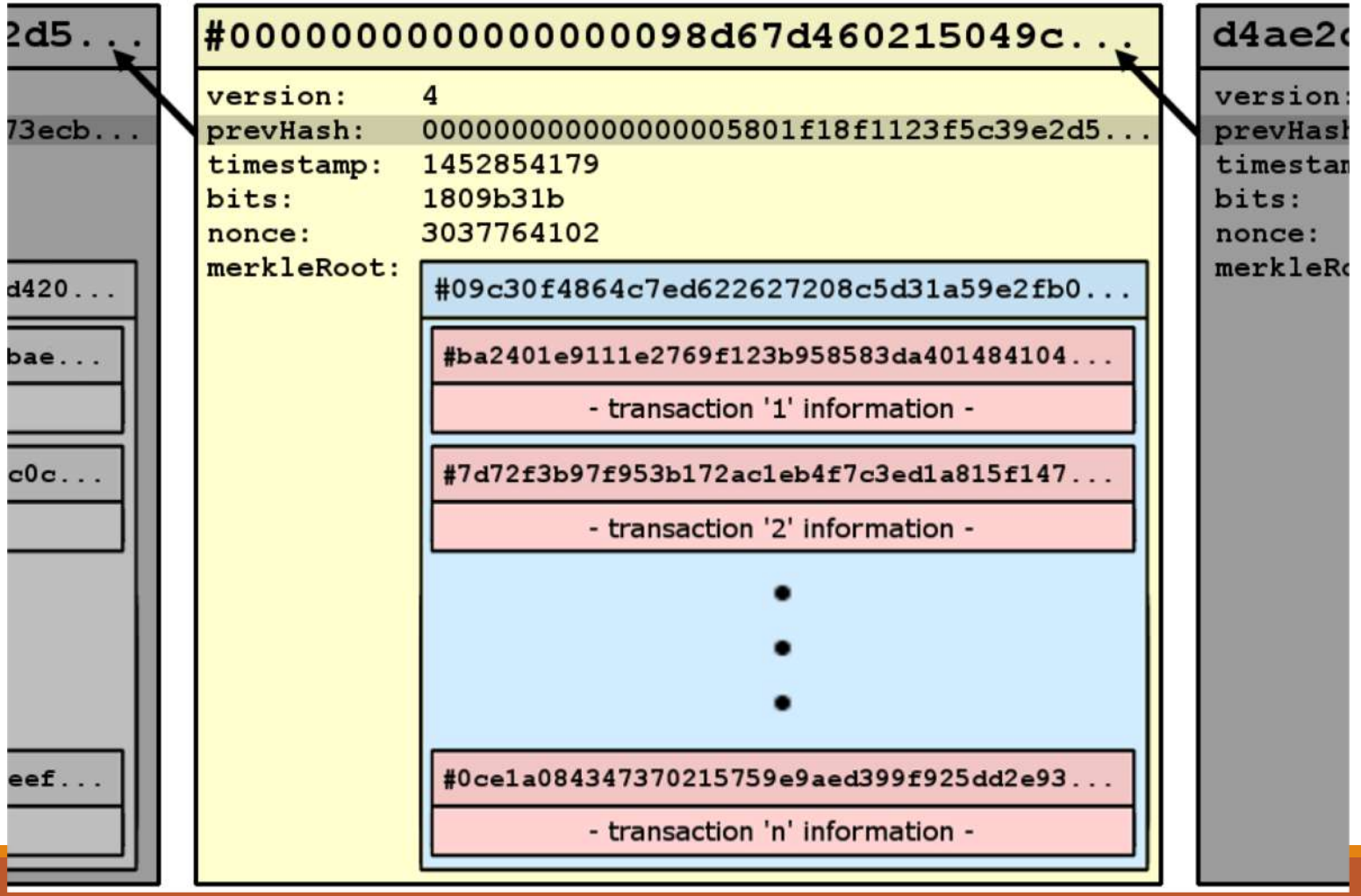
Mining

- Minage = opération de certification d'un bloc
  - Recherche d'une valeur (Nonce) tel que le hash du block (nonce inclu) ait une certaine propriété
    - (1) Bitcoin : commence par un certain nombre de 0

- Bitcoin basé sur le PoW : Proof Of Work
  - Il est très difficile de trouver un block ayant la propriété (1), mais très facile de le vérifier
    - => requiert beaucoup de puissance
    - Chaque block trouvé est remunéré (50 bitcoins puis 25 actuellement puis …)

Exemple en video : https://anders.com/blockchain/

Structure d’un block



Multichaind ?

- Plateforme Ouverte pour les pallications blockchain
  - http://www.multichain.com/
  - Open-source

- Déploiement rapide
- Gestion des permissions
- Nombre de registre illimité
- **Supporte les flux de données**
- Compatible bitcoin

Multichaind : Exemple de commande cli

- Création d'une chaine
  - multichain-util create sensors-data
- Lancement du client sur un nœud
  - multichaind sensors-data –daemon
- Connection d'un client à un autre nœud
  - multichaind sensors-data@[ip-address]:[port]
- Création d'un flux
  - multichain-cli sensors-data create stream temperature false
- Publication d'une donnée dans ce flux
  - multichain-cli sensors-data publish temperature val 22.5
- Minage implicite
  - Les nœuds qui ont le droit de miner pour sensors-data particperont au minage

- **Bien sûr il faut avoir correctement fixé les droits de connection/publication/minage pour chaque noeud**

multichain-web

MultiChain Demo – Default

Node | Permissions | Issue Asset | Update | Send | Create Offer | Accept | Create Stream | Publish | View Streams

My Node

Name	sensors-data
Version	1.0 alpha 28
Protocol	10007
Node address	sensors-data@172.10.0.10:7189
Blocks	34563
Peers	10

Connected Nodes

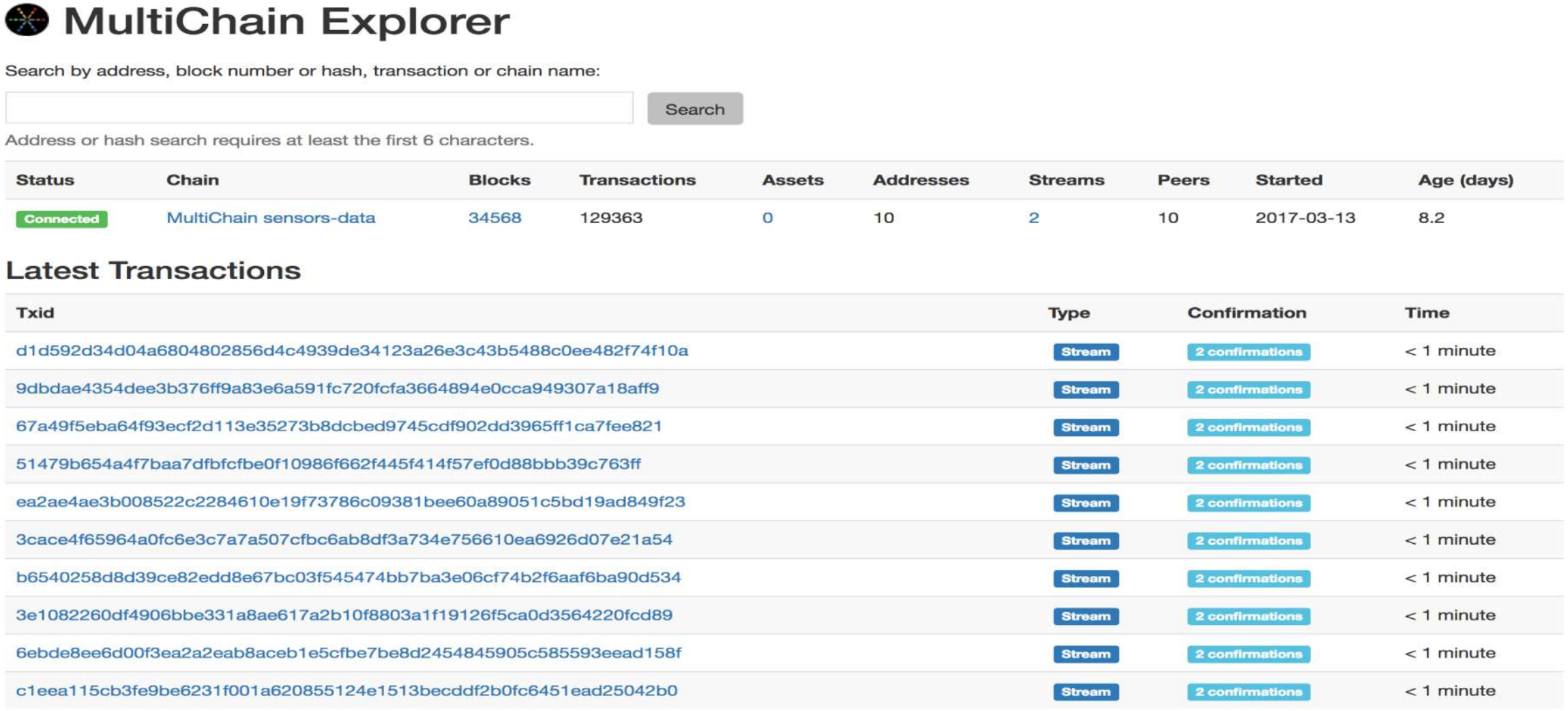
Node IP address	172.10.0.29
Handshake address	
Latency	0.060 sec

My Addresses

Label	Set label
Address	13XAzWv5XLv4f8npwZ7XGJKsLxGLnwfvX5ekoX
Permissions	connect, send, receive, issue, create, mine, admin, activate – change

Get new address

MultiChain Explorer



Block Summary

Hash	00000034484eddb2acf05d831512d28ad77edde9b373361d2a84a4f5ce76f1d9
Previous Block	000000f6267d3e004a32cfab217258145070ec00680da8eca83f999b087ada0c
Next Block	0000001e0c19c56b95bc29c8cc8c9413b4add817ac659e4a22ed4f92baf25d3f
Height	34568
Miner	1Dqczb2dSr7xKXxs37d2vAi1pPhteZcmJrKHuz
Version	3
Transaction Merkle Root	7a9d3584782af6cf1ae314a663a3e797cf50de3baae15d38a15081ad9872b496
Time	1490108170 (2017-03-21 14:56:10)
Nonce	8928364
Transactions	21

Transactions

Transaction	Size (kB)
0e01055ad9f8246f68fc59783094e4ca942f222a1536da75e940a63751854107 Miner Signature	0.187
5a8fc28e8b845e2aa2337bc0bdb0c344c180baa1929f4375169e3dc8fbbfdbae Stream Item	0.235
e7953be4bea791d42608accfebf6530fdbf169e301e0c1d74f8ec1bdf494f453 Stream Item	0.234
749d53c0fb103861c042650da87fda7f41f27a46eb352651c93aa48571492378 Stream Item	0.235

Transaction d1d592d34d...f10a

Hash	d1d592d34d04a6804802856d4c4939de34123a26e3c43b5488c0ee482f74f10a
Appeared in	MultiChain sensors-data, Block 34568 (2017-03-21 14:56:10)
Number of inputs	1 – jump to inputs
Number of outputs	2 – jump to outputs
Size	234 bytes

Bitcoin JSON | MultiChain JSON | MultiChain Hex

Inputs

Index	Previous output	Native	From address	ScriptSig
0	227aaaeecb...:1	0	1LXMmSQpTjs9kSb1VNuGVVANFHQWSZqTewW8FE	71:3044...7d01 33:0342...d2d4

Outputs

Index	Redeemed at input	Native	To address	ScriptPubKey
0	Not yet redeemed	0	None	20:7370...7ee1 DROP 7:7370...616c DROP RETURN 1:29
1	Not yet redeemed	0	1LXMmSQpTjs9kSb1VNuGVVANFHQWSZqTewW8FE	DUP HASH160 20:9073...d086 EQUALVERIFY CHECKSIG

Stream	temperature
Key	val
Data	29

Block	Miner	Approx. Time	Transactions	Chain Age
34577	1Dqczb2dSr7xKXxs37d2vAi1pPhteZcmJrKHuz	2017-03-21 14:58:32	1	8.16243
34576	1LXMmSQpTjs9kSb1VNuGVVANFHQWSZqTewW8FE	2017-03-21 14:58:22	1	8.16231
34575	1YpsMXFEv9m8J51ENKjX8bs9JQheBd9shiABw4	2017-03-21 14:58:05	21	8.16212
34574	1V6xgNRBk1Fm8EUrcAoTusp1692WD5t7r8V3US	2017-03-21 14:57:48	1	8.16192
34573	13XAzWv5XLv4f8npwZ7XGJKsLxGLnwfvX5ekoX	2017-03-21 14:57:37	1	8.16179
34572	1FtbbjBVGaPeL7ssQXKtnBcrkXoCUPRgeQjth2	2017-03-21 14:57:26	1	8.16167
34571	14JLxWu7yCQxBdN8YKzBk9awRAJd5EVue4UZRu	2017-03-21 14:57:00	21	8.16137
34570	1ENh8FUCT6FXuJDtFTggEzEy6DMDpFAdBZJAVC	2017-03-21 14:56:53	1	8.16128
34569	1MLBuRYe6s6fPL1Wq9NoYRci1qSfZ65tLaZsHQ	2017-03-21 14:56:19	1	8.16089
34568	1Dqczb2dSr7xKXxs37d2vAi1pPhteZcmJrKHuz	2017-03-21 14:56:10	21	8.16079
34567	1UHm1knzRdTwWyNvYjGaLp5M4sf3vipQzNZEbf	2017-03-21 14:55:50	1	8.16056
34566	1YpsMXFEv9m8J51ENKjX8bs9JQheBd9shiABw4	2017-03-21 14:55:39	1	8.16043
34565	1V6xgNRBk1Fm8EUrcAoTusp1692WD5t7r8V3US	2017-03-21 14:55:29	1	8.16031
34564	1LXMmSQpTjs9kSb1VNuGVVANFHQWSZqTewW8FE	2017-03-21 14:55:20	21	8.16021

Perspectives / réflexions

Blockchain adaptée au stockage de données issues de l'IoT ?

- Stockage distribué : P2P
- Pas de nœud central
- Imputabilité des données (« preuve »)

Point de blocage ?

- PoW est-il adapté ?
- Stockage des clés ? ***=> dispositif matériel***
- Confidentialité des données ?
- Mode déconnecté ?
- Taille de la chaine ?
- Taille des blocs ?
- Performance ?

Merci de votre attention

Questions ?

Bibliographie

http://www.openstack.org

https://www.slideshare.net/crowdsourcingweek/blockchainsmart-cs2015

https://www.bitcoin.com/

https://www.multichain.com/